

Introduction

1. Overview of the ioapiTools package

The ioapiTools package was developed to include IOAPI data within the CDAT framework. IOAPI is an adaptation of netCDF that is found primarily within the air quality community, specifically in the CMAQ (Community Multiscale Air Quality) model. The ioapiTools module provides functions for extracting and manipulating data (individual variables) and writing to either an IOAPI file or to a climate and forecast compliant (CF) netCDF file with ioapi metadata. For the rest of this document, we will simply refer to these two formats as "IOAPI" and "CF netCDF".

2. ioapiTools and cdms

The ioapiTools module is a contributed package to CDAT and is built on top of the cdms module. The key object in ioapiTools, iovar is a daughter of the cdms transient variable. In other words, an iovar object has all the capabilities of a cdms variable plus some extra methods and attributes. So, you can use the methods and attributes that you may already be familiar with from cdms variables as well as the new methods and attributes.

For those of you who aren't familiar with objects, an object is a way of organizing data and functions together. An object usually has a series of attributes, this is the data (similar to a structure in C). In addition to data, it also may have a series of methods, which are functions that operate on this data.

3. Getting started with python

After you have installed CDAT and the ioapiTools, you should go into a python interpreter to use the packages. There are a whole series of options: python, idle, ipython (my favorite), etc.

So to start from a unix prompt:

```
$ python
Python 2.3.4 (#2, Jan  5 2005, 08:24:51)
[GCC 3.3.5 (Debian 1:3.3.5-5)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
```

This will startup a python interpreter. Unless otherwise noted by "\$", all the following formatted statements are python commands or comments (preceded by a "##")

```
import ioapiTools as iot
```

Here, I've imported the ioapiTools module and said to call it "ioT" instead of the default "ioapiTools".

Two very useful python commands that can be used on any module or object are directory and help:

```
dir(iot)
## Here is the result of directory on iot,
##a long list of data, functions, and classes
['Basemap', 'D', 'MV', 'N', 'NEFlag', 'NWFlag', 'SEFlag', 'SWFlag', '__author__', '__builtins__'
##

## Getting help on open function
help(iot.open)
##
Help on function open in module ioapiTools.ioapiTools:

open(fileName, rwFlag='r', typeFlag=2, newM=None, logFile=None, llaxesFlag=False)
    Opens a file and returns an iofile object.

    Input:
    ...
##
```

For some good introductions to python, try:

- <http://www.hetland.org/python/instant-python.php>
- <http://www.python.org/doc/current/tut/tut.html>

[Contents](#) [Next](#)